AMENDMENTS TO THE CLAIMS

Please amend claims 1-3, 5, 9, 11-18, 21, 22, 24-26, 33, 35, 38, and 40-46, as shown below. Pursuant to 37 C.F.R. § 1.121(c), the text of all pending claims, along with their current status, is set forth below.

1. (Currently Amended) Method, performed by a suitably programmed processor, for software emulation of hard disks of a data processing platform at the level of an operating system with parameterizable management of requests for writing and reading data, ~~consisting of~~ the method comprising:

creating a representation of a real hard disk, wherein the sequence and location for loading and execution of components of the operating system of the data processing platform may be modified,

loading on said data processing platform one or more peripheral drivers, wherein at least one of the peripheral drivers communicates with a data storage support containing the data of the representation of the real hard disk, and

simulating behavior of the real hard disk for the operating system, wherein the method transforms programming contained on the real hard disk into an emulated hard disk capable of controlling read and write operations on a client station and among two or more client stations.

2. (Currently Amended) Method as claimed in claim 1, wherein the management of said data write requests that the operating system sends to the emulated hard disk is accomplished at [[the]] a peripheral driver level and/or at [[the]] a level of an optional hard disk server service on a data processing network, the written data being stored according to the parameterization of said peripheral drivers and/or said service server of the hard disk on the network, and wherein the parameterization accommodates the written data being stored in any one of:

- ~~either directly in~~ the support containing the emulated hard disk[[,]];

- [[or]] ~~in~~ the memory, random access or virtual, accessible to the operating system using the emulated hard disk[[,]];
- [[or]] ~~in~~ a volatile storage space accessible to the operating system using the emulated hard disk[[,]];
- [[or]] ~~in~~ a non-volatile storage space accessible to the operating system using the emulated hard disk[[,]];
- [[or]] ~~in~~ a volatile storage space accessible to the server service of emulated hard disks on a data processing network[[,]]; or
- [[or]] ~~in~~ a non-volatile storage space accessible to the server service of emulated hard disks on a data processing network.

3. (Currently Amended) Method as claimed in claim 1, wherein the management of the data reading requests that the operating system issues to the emulated hard disk is accomplished at [[the]] a peripheral driver level and/or at [[the]] a level of an optional hard disk server service on a data processing network, the readings of previously written data being performed in the storage space of any one of:

- ~~either directly in~~ the support containing the emulated hard disk[[,]];
- ~~or in the~~ random access or virtual memory accessible to the operating system using the emulated hard disk[[,]];
- ~~or in~~ a volatile storage space accessible to the operating system using the emulated hard disk[[,]];
- ~~or in~~ a nonvolatile storage space accessible to the operating system using the emulated hard disk[[,]];
- ~~or in~~ a volatile storage space accessible to the server service of emulated hard disks on a data processing network[[,]]; or
- ~~or in~~ a non-volatile storage space accessible to the server service of emulated hard disks on a data processing network.

4. (Previously Presented) Method as claimed in claim 1, wherein the emulation of the hard disk provided to the operating system of the client station is accomplished by the agency of a single, monolithic peripheral driver which communicates with the operating system in the manner of a hard disk and which communicates with the support containing the data of said emulated hard disk in a manner specific to this support.

5. (Currently Amended) Method as claimed in claim 1, wherein the data of the emulated hard disk ~~or disks~~ are accessible to the client stations via a data processing network, and wherein the emulated hard disk comprises one or more emulated hard disks.

6. (Previously Presented) Method as claimed in claim 1, wherein when an emulated hard disk is started up, the method further comprises providing a low level micro-software module to access the data contained in the emulated hard disk, wherein the operating system is started up at the client station.

7. (Previously Presented) Method as claimed in claim 6, wherein the data processing network comprises a plurality of client stations, the client stations using a bootup PROM, the method further comprising using the bootup PROM to control communications via the data processing network.

8. (Previously Presented) Method as claimed in claim 7, wherein the low level micro-software module is loaded in the memory of the client station and executed using the bootup PROM.

9. (Currently Amended) Method as claimed in claim 6, wherein the low level micro-software module is loaded in memory of the client station and executed as a component of the BIOS of the client station, said low level micro-software module providing the same functions as [[the]] access services on real hard disks provided by the BIOS.

10. (Previously Presented) Method as claimed in claim 6, wherein the low-level micro-software is loaded in memory of the client station from a third party data support supported as a startup peripheral by the client station.

11. (Currently Amended) Method as claimed in claim 5, wherein at least one peripheral driver loaded and executed by the operating system of the client station provides the functions of access, via the data processing network, to the data contained in the emulated hard disk[[s]].

12. (Currently Amended) Method as claimed in claim 1, wherein if the data support containing the data of the emulated hard disk[[(s)]] is a support that does not provide for writing in real time, or does not accept writing of data directly in the support containing the data of the emulated hard disk, the data writing requests issued by the operating system to the emulated hard disk[[(s)]] are processed in such a way that the written data are stored in a storage space different from the data support containing the data of the emulated hard disk[[(s)]], wherein the emulated hard disk comprises one or more emulated hard disks.

13. (Currently Amended) Method as claimed in claim 12, wherein the data writing requests issued by the client station operating system to the emulated hard disk[[(s)]] are processed in such a way that the written data are stored in [[the]] a random access memory of the client station.

14. (Currently Amended) Method as claimed in claim 12, wherein the data writing requests issued by the client station operating system to the emulated hard disk[[(s)]] are processed in such a way that the written data are stored in [[the]] a virtual memory of the client station.

15. (Currently Amended) Method as claimed in claim 12, wherein the data writing requests issued by the client station operating system to the emulated hard disk[[(s)]] are processed in such a way that the written data are stored in a data file accessible to the operating system of the client station.

16. (Currently Amended) Method as claimed in claim 1, wherein the data writing requests issued by the operating system to the emulated hard disk[[(s)]] are redirected to a single storage space, and wherein the storage space in which the written data are redirected may be changed during an operating session of the operating system of a client station.

17. (Currently Amended) Method as claimed in claim 12, wherein the storage space used for storage of the written data may be volatile, or may be nonvolatile so as to permit the written data of an operating session of the operating system to persist from one client station to another.

18. (Currently Amended) Method as claimed in claim l6, wherein [[the]] a volatile character of the redirections of [[the]] written data is determined upon initialization of the operating session of the operating system of the client station.

19. (Previously Presented) Method as claimed in claim 1, wherein the data reading requests issued by the operating system are performed in different storage spaces during an operating session of the operating system of a client station.

20. (Previously Presented) Method as claimed in claim 19, wherein the data reading requests issued by the operating system to an emulated hard disk carried out in different storage spaces follow an order of priority.

21. (Currently Amended) Method as claimed in claim 5, wherein a server program is in charge at one of the client stations of the data processing network, on the one hand, of the communications via the data processing network with the client stations accessing the emulated hard disks, and on the other, of accessing the data support containing the data of the emulated hard disk[[s]].

22. (Currently Amended) Method as claimed in claim 21, wherein if the hard disk emulation ~~system~~ is parameterized so that the data write requests received by the server program are intended for a specific emulated hard disk they are not redirected but stored directly in a support containing the data of the emulated hard disk itself, and only one client station can access said emulated hard disk at a given time.

23. (Previously Presented) Method as claimed in claim 21, wherein in order to permit several client stations to access an emulated hard disk simultaneously, the server program is capable of redirecting specifically the data write requests issued by a client station A to a given storage space, and of redirecting the data write requests issued by another client station B to another given storage space.

24. (Currently Amended) Method as claimed in claim [[1]] 21, wherein in order to permit [[the]] startup from and/or simultaneous access to the same emulated hard disk or 100% identical copies of the same emulated hard disk, components of the operating system loaded and executed by the client stations or server program are capable of modifying, during or before their use by the operating system, data contained in the emulated hard disk.

25. (Currently Amended) Method as claimed in claim 1, wherein [[the]] emulation is performed for the operating system of the client stations at a level of a class of virtual peripherals of a file system type.

26. (Currently Amended) Method as claimed in claim 1, wherein [[the]] emulation is performed for the operating system of the client stations at the level of the class of disk peripherals itself and not at [[the]] a file system level.

27. (Previously Presented) Method as claimed in claim 1, wherein data contained in the emulated hard disk are copied by a software tool executed at a client station from the real hard disk.

28. (Previously Presented) Method as claimed in claim 27, wherein the software tool creates an image directory that contains the data of the emulated hard disk.

29. (Previously Presented) Method as claimed in claim 27, wherein the software tool creates an image file that contains the data of the emulated hard disk.

30. (Previously Presented) Method as claimed in claim 1, wherein in order to permit startup from an emulated hard disk, the sequence of loading of the components of the operating system is modified so that all components of the operating system on which the peripheral drivers permitting access to the emulated hard disk depend are loaded and usable at the moment when the operating system accesses the emulated hard disk by using the peripheral drivers.

31. (Previously Presented) Method as claimed in claim 21, wherein in order to accelerate the simultaneous access by several client stations to the same emulated hard disk whose data are contained in a data support accessible to a server station, the data are sent by the server station to the client stations using broadcast or multicast mechanisms.

32. (Previously Presented) Method as claimed in claim 31, wherein the data sent by broadcast or by multicast by the server station are stored by the client stations that accept them in a local cache situated in the memory of said client stations.

33. (Currently Amended) Method as claimed in claim 31, wherein a read request for data in the emulated hard disk issued by the operating system of a client station generates an data reading request sent to the server station only if said data are not already present in [[said]] local cache.

34. (Previously Presented) Method as claimed in claim 33, wherein the data read in the local cache are removed after being read by the client station so as to free up space in said local cache.

35. (Currently Amended) Method as claimed in claim 31, wherein [[the]] a decision to send data by "multicast/broadcast" is made at [the] a server module level which provides ~~the~~ functionalities ~~necessary~~ for the hard disk emulation at the client stations.

36. (Previously Presented) Method as claimed in claim 31, wherein the client stations may modify their subscription to receiving the data sent via "broadcast/multicast" by the server station.

37. (Previously Presented) Method as claimed in claim 32, wherein the client stations may erase the data from the local cache after a certain parameterizable time.

38. (Currently Amended) Method as claimed in claim 5, wherein data contained in ~~the emulated the~~ a server module making the hard disks available to client stations may use any suitable network protocol including one of NVD, iSCSI, and SMB/CFIS.

39. (Previously Presented) Method as claimed in claim 5, wherein a low level software program executed by the client stations permits access to the data contained in the emulated hard disks using any suitable network protocol including one of NVD, iSCSI, and SMB/CFIS.

40. (Currently Amended) Method as claimed in claim 11, wherein [[the]] at least one peripheral driver[[(s)]] executed by the client stations permit access to the data contained in the emulated hard disks any suitable network protocol including one of NVD, iSCSI, and SMB/CFIS.

41. (Currently Amended) Method as claimed in claim 21, wherein if the data storage support containing the data of the emulated hard disk[[(s)]] is a support that does not provide writing in real time, or does not accept write operations directly in the support containing the data of the emulated hard disk, the server program providing the emulation of the hard disk at the client stations processes the data write requests issued by the operating system to the emulated hard disk[[(s)]] in such a way that the written data are stored in a storage space different from the data storage support containing the data of the emulated hard disk[[(s)]].

42. (Currently Amended) Method as claimed in claim 21, wherein the data write requests issued by the client station operating system to the emulated hard disk[[(s)]] are processed in such a way that the written data are stored in [[the]] a random access memory of [[the]] a server station.

43. (Currently Amended) Method as claimed in claim 21, wherein the data write requests issued by the client station operating system to the emulated hard disk[[(s)]] are processed in such a way that the written data are stored in [[the]] a virtual memory of [[the]] a server station.

44. (Currently Amended) Method as claimed in claim 21, wherein the data write requests issued by the client station operating system to the emulated hard disk[[(s)]] are processed in such a way that the written data are stored in a data tile accessible to [[the]] a server software.

45. (Currently Amended) Method as claimed in claim 21, wherein the storage space used for storage of the written data may be volatile, or may be nonvolatile so as to permit the written data of an operating session of the operating system to persist from one client station to another.

46. (Currently Amended) Method as claimed in claim 16, wherein [[the]] a volatile character of the redirections of the written data is determined upon initialization of the operating session of the operating system of the client station.